Buku Saku Arduino

Kafemikrochip

2025

Kafemikrochip

Kafemikrochip

Halaman Hak Cipta

Judul Buku	Panduan Mempelajari Arduino Tingkat Dasar
Pengarang	Eko Travada Suprapto Putro, S.T., M.T.
Tahun Terbit	2025
Edisi	Edisi Pertama
Penerbit	KafeMikrochip
Situs Web	Kafemikrochip.com

Lisensi Penggunaan

Daftar Isi

Daftar Gambar

Panduan Mempelajari Arduino Tingkat Dasar

Pengantar

Selamat datang di dunia Arduino! Panduan ini dirancang untuk membantu Anda, para pemula, memahami konsep dasar dan memulai perjalanan Anda dalam membuat proyek elektronik interaktif menggunakan platform Arduino. Arduino adalah platform open-source yang luar biasa, terdiri dari papan mikrokontroler (hardware) dan perangkat lunak (IDE) yang mudah digunakan, memungkinkan siapa saja, dari seniman hingga insinyur, untuk mewujudkan ide-ide kreatif mereka.

Panduan ini akan membawa Anda langkah demi langkah, mulai dari pengenalan board Arduino itu sendiri, komponen elektronik dasar yang sering digunakan, dasar-dasar pemrograman dengan bahasa C/C++ yang disederhanakan, hingga implementasi praktis mengontrol LED, membaca tombol, menggunakan sensor umum (suhu, cahaya, jarak), dan menampilkan informasi ke layar LCD. Di akhir panduan, terdapat soal uji kompetensi untuk mengukur pemahaman Anda.

Mari kita mulai petualangan seru ini dan ubah ide Anda menjadi kenyataan dengan Arduino!

Bab 1: Mengenal Board Arduino

Selamat datang di dunia Arduino! Jika Anda baru memulai perjalanan dalam elektronika dan pemrograman, Arduino adalah titik awal yang fantastis. Platform ini dirancang khusus untuk memudahkan siapa saja, mulai dari seniman, desainer, penghobi, hingga insinyur, untuk menciptakan berbagai proyek interaktif, mulai dari robot sederhana hingga sistem otomatisasi rumah yang kompleks. Bab ini akan memperkenalkan Anda pada konsep dasar Arduino, komponen-komponen utama yang ada di boardnya, berbagai jenis board yang tersedia, serta perangkat lunak yang digunakan untuk memprogramnya.

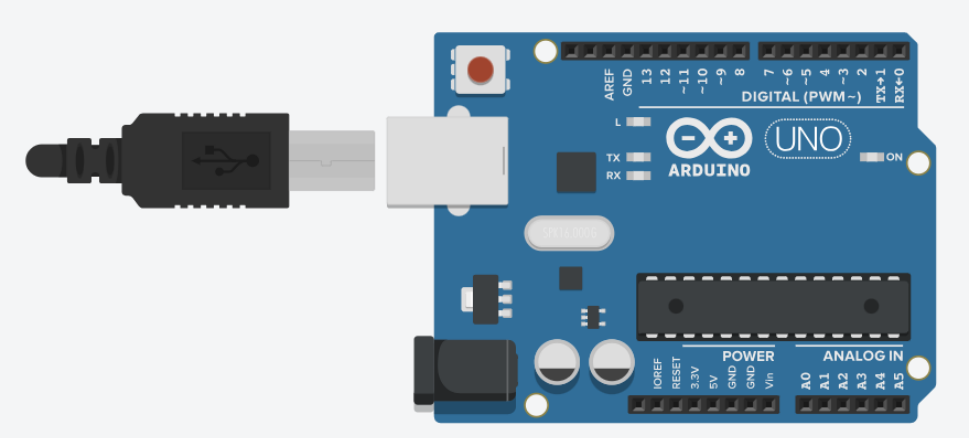
Apa Itu Arduino?

Arduino adalah sebuah platform *prototyping* elektronik yang bersifat *open-source* (sumber terbuka). Ini berarti desain perangkat keras (hardware) dan perangkat lunak (software) Arduino dapat diakses, dimodifikasi, dan didistribusikan secara bebas oleh siapa saja. Inti dari platform ini adalah kombinasi antara papan sirkuit fisik (yang sering disebut sebagai "board Arduino") yang berisi mikrokontroler, dan perangkat lunak berupa Lingkungan Pengembangan Terpadu atau *Integrated Development Environment* (IDE) yang berjalan di komputer Anda. IDE inilah yang digunakan untuk menulis dan mengunggah kode program ke board Arduino.

Fleksibilitas dan kemudahan penggunaan adalah kunci utama popularitas Arduino. Tujuannya adalah menyederhanakan proses pembuatan prototipe elektronik, sehingga individu tanpa latar belakang teknis yang mendalam pun dapat mulai bereksperimen dan mewujudkan ide-ide kreatif mereka. Didukung oleh komunitas global yang besar dan aktif, pengguna Arduino dapat dengan mudah menemukan berbagai macam contoh proyek, tutorial, dan pustaka kode (*library*) yang siap pakai untuk mempercepat pengembangan.

Komponen Utama Board Arduino (Contoh: Arduino UNO)

Meskipun ada banyak jenis board Arduino, sebagian besar memiliki komponen inti yang serupa. Mari kita lihat komponen utama pada Arduino UNO, salah satu board yang paling populer dan direkomendasikan untuk pemula:



Gambar 1 Arduino Uno (R3)

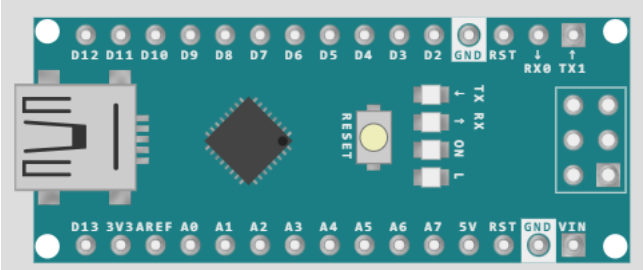
(ThinkerCad, 2025)

1. Mikrokontroler: Ini adalah "otak" dari board Arduino. Pada Arduino UNO R3, mikrokontroler yang digunakan adalah ATmega328P. IC (Integrated Circuit) kecil inilah yang menjalankan kode program yang Anda unggah, membaca input, dan mengontrol output.
2. Pin Input/Output (I/O): Ini adalah pin-pin di sepanjang tepi board yang memungkinkan Arduino berinteraksi dengan dunia luar.
   a. Pin Digital (0-13): Dapat berfungsi sebagai input (membaca sinyal digital HIGH/LOW, seperti dari tombol) atau output (mengirim sinyal digital HIGH/LOW, seperti untuk menyalakan LED). Beberapa pin digital (ditandai dengan '~') juga mendukung *Pulse Width Modulation* (PWM), yang memungkinkan output analog semu (misalnya untuk mengatur kecerahan LED atau kecepatan motor).
   b. Pin Analog Input (A0-A5): Digunakan khusus untuk membaca sinyal analog (tegangan yang bervariasi, bukan hanya HIGH/LOW), seperti dari sensor suhu atau cahaya (LDR). Pin ini memiliki konverter Analog-ke-Digital (ADC) internal.
3. Pin Power: Menyediakan berbagai level tegangan untuk memberi daya pada komponen eksternal.
   c. VIN: Pin untuk memberikan daya eksternal ke Arduino (selain melalui USB atau colokan DC).
   d. 5V: Output tegangan 5 Volt yang sudah diregulasi, umum digunakan untuk memberi daya pada sensor dan komponen lain.
   e. 3.3V: Output tegangan 3.3 Volt yang sudah diregulasi, dibutuhkan oleh beberapa sensor dan modul.
   f. GND (Ground): Pin referensi 0 Volt. Semua komponen dalam satu sirkuit harus terhubung ke ground yang sama.
   g. IOREF: Menyediakan referensi tegangan operasi I/O board (misal 5V untuk UNO).
4. Konektor USB: Port tipe B (seperti pada printer) yang digunakan untuk menghubungkan Arduino ke komputer. Koneksi ini berfungsi untuk mengunggah program dan menyediakan daya ke board, serta untuk komunikasi serial antara Arduino dan komputer.
5. Colokan Daya DC (DC Power Jack): Memungkinkan Arduino ditenagai oleh sumber daya eksternal (adaptor AC-ke-DC atau baterai) dengan rentang tegangan biasanya 7-12 Volt.
6. Regulator Tegangan: Komponen yang mengambil tegangan dari colokan DC atau pin VIN dan menurunkannya menjadi tegangan stabil 5V dan 3.3V yang dibutuhkan oleh mikrokontroler dan pin output.
7. Tombol Reset: Tombol kecil untuk memulai ulang eksekusi program dari awal (sama seperti mencabut dan memasang kembali kabel USB).
8. LED Bawaan (Built-in LED): LED kecil yang terhubung ke salah satu pin digital (biasanya pin 13 pada UNO). Sangat berguna untuk pengujian dasar program tanpa perlu merangkai komponen eksternal.
9. LED Indikator (TX, RX, Power): LED kecil yang menunjukkan status daya (ON) dan aktivitas komunikasi serial (TX - Transmit, RX - Receive).
10. Osilator Kristal (Crystal Oscillator): Menghasilkan sinyal detak (clock signal) yang presisi (biasanya 16 MHz pada UNO) untuk mengatur kecepatan eksekusi instruksi oleh mikrokontroler.

Macam-macam Board Arduino

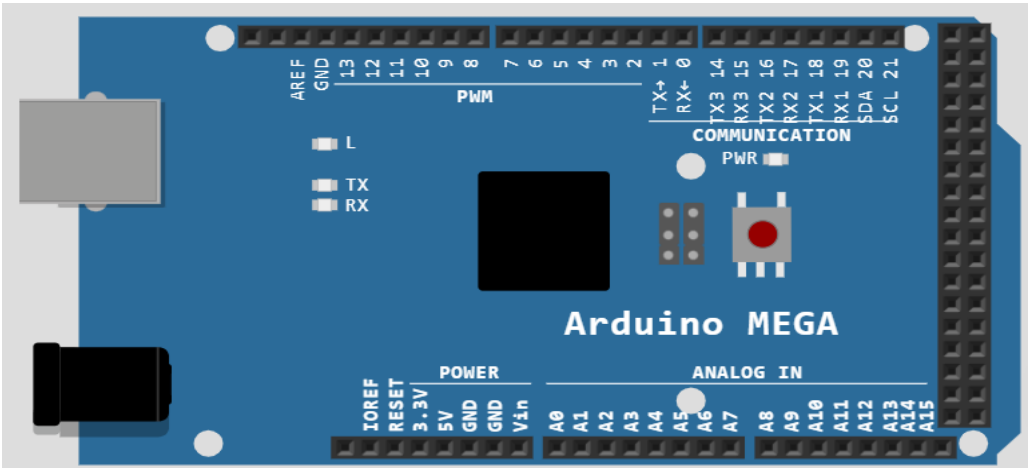
Arduino hadir dalam berbagai bentuk dan ukuran, masing-masing dengan fitur dan spesifikasi yang berbeda untuk memenuhi kebutuhan proyek yang beragam. Berikut beberapa keluarga dan board Arduino yang populer:

1. Keluarga Classic (Contoh: Arduino UNO R3): Tampak pada gambar 1.1. Ini adalah board "standar" dan paling ikonik. Arduino UNO R3 sangat direkomendasikan untuk pemula karena keseimbangan antara jumlah pin, kemudahan penggunaan, dan banyaknya sumber belajar yang tersedia. Board lain dalam keluarga ini termasuk Leonardo (dengan kemampuan USB HID bawaan) dan Micro (versi lebih kecil dari Leonardo).
2. Keluarga Nano: Board dengan ukuran sangat kecil (*small footprint*), cocok untuk proyek yang membutuhkan dimensi ringkas. Contohnya termasuk Nano Every (versi dasar yang terjangkau), Nano 33 IoT (dengan konektivitas Wi-Fi), Nano 33 BLE Sense (dengan Bluetooth® Low Energy dan berbagai sensor terintegrasi seperti suhu, kelembaban, tekanan, gestur, mikrofon), dan Nano RP2040 Connect (menggunakan chip Raspberry Pi RP2040 dengan Wi-Fi/Bluetooth®).



Gambar 2 Arduino Nano (Wokwi, 2019)

3. Keluarga Mega (Contoh: Arduino Mega 2560 Rev3): Dirancang untuk proyek yang lebih kompleks yang membutuhkan jumlah pin I/O yang jauh lebih banyak (lebih dari 50 pin digital) dan memori yang lebih besar dibandingkan UNO. Arduino Due juga termasuk dalam kategori ini, menggunakan mikrokontroler berbasis ARM 32-bit yang lebih bertenaga (namun beroperasi pada 3.3V).

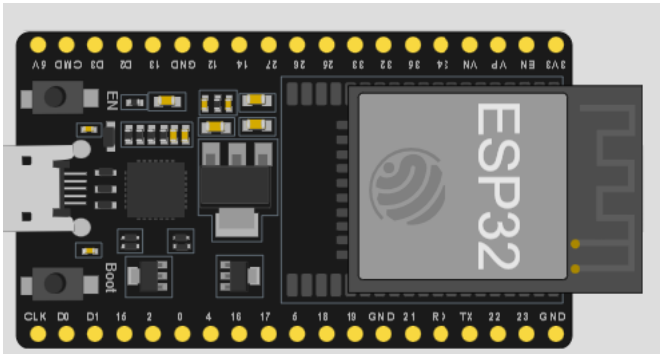


Gambar 3 *Arduino Mega (Wokwi, 2019)*

4. Keluarga MKR: Rangkaian board modular yang dirancang khusus untuk proyek IoT (Internet of Things) dan konektivitas. Board MKR biasanya
5. dilengkapi dengan modul radio terintegrasi (seperti Wi-Fi, Bluetooth®, LoRa®, Sigfox®, NB-IoT) dan chip kripto untuk komunikasi yang aman. Tersedia juga berbagai *shield* dan *carrier* MKR untuk menambah fungsionalitas dengan mudah.

Gambar 4 Board MKR *(Arduino, 2025)*

6. Board Berbasis ESP: Meskipun tidak secara resmi di bawah merek "Arduino", board seperti ESP8266 (NodeMCU) dan ESP32 sangat populer di kalangan komunitas Arduino karena harganya yang murah, memiliki Wi-Fi dan Bluetooth® terintegrasi, serta dapat diprogram menggunakan Arduino IDE setelah menginstal dukungan board tambahan.



Gambar 5 Board ESP 32 *(Wokwi, 2019)*

Selain board resmi dari Arduino, ada juga board *clone* (tiruan tidak resmi yang seringkali lebih murah) dan board *compatible* (dibuat oleh produsen lain tetapi dirancang agar kompatibel dengan ekosistem Arduino). Meskipun board clone bisa menjadi alternatif yang lebih ekonomis, kualitas dan kompatibilitasnya terkadang bervariasi.

Memilih Board Arduino yang Tepat untuk Pemula

Untuk pemula, Arduino UNO R3 (atau clone berkualitas baik) umumnya merupakan pilihan terbaik. Alasannya:

- Populer: Paling banyak digunakan, sehingga tutorial, contoh proyek, dan dukungan komunitas sangat melimpah.

- Ukuran Standar: Ukurannya cukup besar untuk mudah ditangani, dan kompatibel dengan sebagian besar *shield* (papan tambahan yang dipasang di atas Arduino untuk menambah fungsi).
- Jumlah Pin Cukup: Memiliki jumlah pin I/O yang memadai untuk sebagian besar proyek pemula.
- Tegangan Operasi 5V: Banyak sensor dan modul pemula dirancang untuk beroperasi pada 5V.
- Harga Terjangkau: Relatif tidak mahal, terutama jika mempertimbangkan versi clone.

Alternatif lain yang baik untuk pemula adalah Arduino Nano, yang memiliki fungsionalitas mirip UNO tetapi dalam ukuran yang jauh lebih kecil, membuatnya cocok untuk dipasang langsung di breadboard.

Pengenalan Arduino IDE

Arduino IDE (Integrated Development Environment) adalah perangkat lunak gratis yang Anda instal di komputer (Windows, macOS, atau Linux) untuk menulis, mengompilasi (*compile*), dan mengunggah (*upload*) kode program ke board Arduino.

Instalasi:

- Unduh versi terbaru dari situs web resmi Arduino (www.arduino.cc).
- Ikuti petunjuk instalasi untuk sistem operasi Anda. Proses ini biasanya juga akan menginstal driver USB yang diperlukan agar komputer dapat mengenali board Arduino.

Antarmuka Utama:

- Area Kode (Text Editor): Tempat Anda menulis kode program (disebut *sketch* dalam terminologi Arduino).
- Tombol Verify (✓): Mengompilasi kode Anda untuk memeriksa kesalahan sintaks tanpa mengunggahnya ke board.
- Tombol Upload (→): Mengompilasi kode Anda dan, jika tidak ada error, mengunggahnya ke board Arduino yang terhubung.
- Nama Sketch: Menampilkan nama file sketch yang sedang dibuka.
- Area Pesan (Message Area): Menampilkan pesan status selama kompilasi dan unggah, serta pesan kesalahan jika ada.
- Konsol Output (Output Console): Menampilkan output yang lebih detail dari proses kompilasi dan unggah.
- Menu Board & Port (Tools > Board & Tools > Port): Tempat Anda memilih jenis board Arduino yang Anda gunakan dan port serial COM tempat board terhubung ke komputer. Penting: Pastikan Anda memilih board dan port yang benar sebelum mengunggah!

- Serial Monitor (Tombol di pojok kanan atas atau Tools > Serial Monitor): Membuka jendela terpisah yang memungkinkan komunikasi dua arah antara Arduino dan komputer melalui koneksi USB/Serial. Sangat berguna untuk menampilkan data dari Arduino (misalnya nilai sensor) atau mengirim perintah ke Arduino.

Gambar 6 Arduino IDE *(Arduino, 2025)*

Dengan memahami komponen dasar board dan cara kerja IDE, Anda kini siap untuk melangkah ke bab berikutnya, di mana kita akan membahas dasar-dasar elektronika yang diperlukan untuk mulai merangkai sirkuit sederhana dengan Arduino Anda.

Referensi:

- https://www.arduino.cc/en/hardware/
- https://fikom.udb.ac.id/artikel/detail/komponen-arduino-dan-fungsinya

---

Bab 2: Elektronika Dasar Digital untuk Arduino

Setelah mengenal board Arduino dan IDE-nya, langkah selanjutnya adalah memahami beberapa komponen elektronika dasar yang akan sering Anda gunakan dalam proyek-proyek Arduino. Bab ini akan membahas konsep dasar kelistrikan, pengenalan breadboard sebagai tempat merangkai, serta fungsi dan cara kerja komponen seperti resistor, LED, push button, dan transistor.

Pengantar Singkat Arus, Tegangan, dan Resistansi (Hukum Ohm)

Sebelum merangkai komponen, penting untuk memahami tiga konsep dasar dalam kelistrikan:

1. Tegangan (Voltage - V): Bisa dibayangkan sebagai "dorongan" atau "tekanan" yang menyebabkan muatan listrik (elektron) mengalir. Satuan tegangan adalah Volt (V). Arduino UNO umumnya beroperasi pada 5V.
2. Arus (Current - I): Adalah aliran muatan listrik itu sendiri. Satuan arus adalah Ampere (A), namun dalam proyek Arduino seringkali menggunakan satuan yang lebih kecil seperti milliampere (mA) (1A = 1000mA).
3. Resistansi (Resistance - R): Adalah hambatan terhadap aliran arus listrik. Komponen yang memiliki resistansi disebut resistor. Satuan resistansi adalah Ohm (Ω). Semakin besar resistansi, semakin sulit arus mengalir.

Ketiga konsep ini saling berhubungan melalui Hukum Ohm, yang menyatakan:

$$V = I * R$$

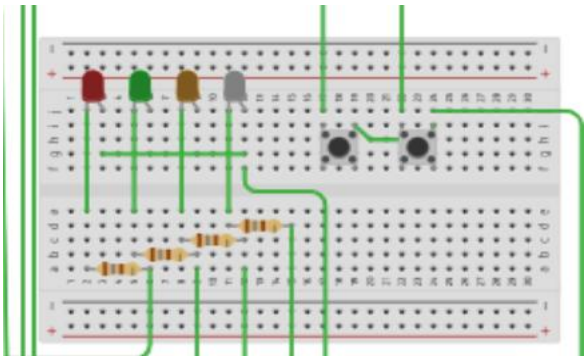
Artinya, tegangan (V) sebanding dengan hasil kali antara arus (I) dan resistansi (R). Hukum ini sangat penting untuk menghitung nilai komponen yang tepat, misalnya menghitung resistor yang dibutuhkan untuk sebuah LED.

Breadboard: Tempat Merangkai Tanpa Solder

Breadboard (atau sering disebut project board) adalah papan plastik dengan banyak lubang kecil yang memungkinkan Anda merangkai komponen elektronik dan menghubungkannya dengan kabel jumper tanpa perlu menyolder. Ini sangat ideal untuk prototyping dan eksperimen.

Cara Kerja Breadboard:

- Lubang Terhubung: Lubang-lubang pada breadboard terhubung secara internal oleh klip logam.
- Baris Terminal (Bagian Tengah): Lubang-lubang dalam satu baris pendek (biasanya 5 lubang) di bagian tengah breadboard saling terhubung secara horizontal. Baris yang berbeda tidak terhubung.



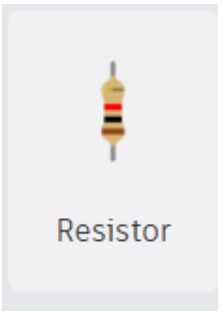
Gambar 7 Breaboard *(ThinkerCad, 2025)*

Jalur Daya (Bus Strips - Bagian Tepi): Biasanya terdiri
- atas dua kolom di setiap sisi breadboard (ditandai '+' dan '-') yang lubangnya terhubung secara vertikal sepanjang kolom tersebut. Ini digunakan untuk mendistribusikan jalur daya (VCC/+5V) dan ground (GND/-) ke seluruh rangkaian.
- Pemisah Tengah (Central Notch/Divider): Celah di tengah memisahkan dua bagian baris terminal, cocok untuk memasang IC (Integrated Circuit) agar pin di kedua sisinya tidak saling terhubung.

Dengan breadboard, Anda bisa dengan mudah memasang komponen, menghubungkannya dengan kabel jumper, dan mengubah rangkaian jika diperlukan.

Resistor: Si Penahan Arus

Resistor adalah komponen pasif yang fungsi utamanya adalah untuk menahan atau membatasi aliran arus listrik dalam sebuah rangkaian. Nilai kemampuannya menahan arus disebut resistansi, diukur dalam Ohm (Ω).



Fungsi Utama Resistor dalam Proyek Arduino:
1. Pembatas Arus: Fungsi paling umum. Misalnya, LED membutuhkan arus tertentu untuk menyala terang tetapi akan rusak jika arusnya terlalu besar. Resistor dipasang seri dengan LED untuk membatasi arus yang mengalir melaluinya sesuai dengan Hukum Ohm.
2. Resistor Pull-up / Pull-down: Digunakan bersama komponen input seperti tombol (push button) untuk memastikan pin input Arduino selalu mendapatkan sinyal HIGH atau LOW yang pasti saat tombol tidak ditekan, mencegah kondisi *floating* (akan dibahas lebih lanjut di bagian Push Button).

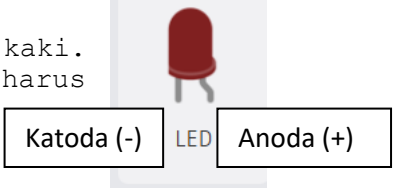
3. Pembagi Tegangan (Voltage Divider): Dua resistor yang dipasang seri dapat digunakan untuk menurunkan tegangan sumber menjadi level tegangan yang lebih rendah.

Membaca Nilai Resistor (Kode Warna - Opsional):
Resistor fisik biasanya memiliki gelang-gelang warna yang menandakan nilai resistansinya dan toleransinya. Anda dapat mencari "kalkulator kode warna resistor" online untuk mengetahui cara membacanya. Namun, untuk pemula, seringkali lebih mudah menggunakan multimeter untuk mengukur nilai resistansi atau membeli kit resistor yang sudah diberi label nilainya.

LED: Si Pemberi Cahaya

LED (Light Emitting Diode) adalah komponen semikonduktor yang memancarkan cahaya ketika dialiri arus listrik. LED adalah komponen output yang paling umum digunakan dalam proyek Arduino pemula.

Karakteristik LED:



- Dioda: Arus hanya bisa mengalir dalam satu arah.
- Kaki Anoda (+) dan Katoda (-): LED memiliki dua kaki. Kaki yang lebih panjang adalah Anoda (+), yang harus terhubung ke sisi tegangan positif (misalnya pin output Arduino atau VCC). Kaki yang lebih pendek adalah Katoda (-), yang harus terhubung ke sisi tegangan negatif (Ground/GND).
- Tegangan Maju (Forward Voltage - Vf): Tegangan minimum yang dibutuhkan LED untuk mulai menyala (bervariasi tergantung warna, biasanya sekitar 1.8V - 3.3V).
- Arus Maju (Forward Current - If): Arus ideal agar LED menyala terang dan awet (biasanya sekitar 10mA - 20mA).

Penting: Menggunakan Resistor Pembatas Arus!
LED tidak bisa langsung dihubungkan antara pin output Arduino dan GND. Pin output Arduino (5V) jauh lebih tinggi dari Vf LED, dan tanpa pembatas, arus yang mengalir akan sangat besar dan merusak LED (atau bahkan pin Arduino). Oleh karena itu, selalu pasang resistor secara seri dengan LED. Nilai resistor yang umum digunakan untuk LED standar dengan Arduino 5V adalah antara 220Ω hingga 1kΩ (misalnya 330Ω atau 470Ω).

Perhitungan Resistor (Contoh):

- Jika V_sumber = 5V (dari pin Arduino), Vf_LED = 2V, dan If_LED = 15mA (0.015A), maka:
- Tegangan yang perlu "dibuang" oleh resistor (Vr) = V_sumber - Vf_LED = 5V - 2V = 3V.
- Nilai Resistor (R) = Vr / If_LED = 3V / 0.015A = 200Ω. (Gunakan nilai standar terdekat yang lebih besar, misal 220Ω).

Push Button: Memberi Input Sederhana

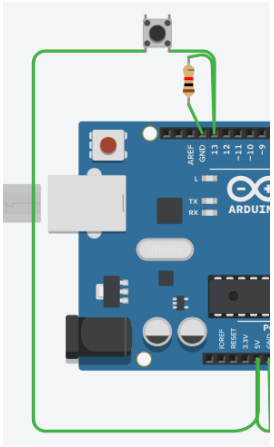
Push button (tombol tekan) adalah saklar sesaat (*momentary switch*) yang paling sederhana. Ia menghubungkan rangkaian saat ditekan dan memutuskannya saat dilepas. Ini cara mudah bagi pengguna untuk memberikan input ke Arduino.

Cara Kerja: Saat ditekan, dua kontak internal terhubung.

Masalah: Floating Pin
Jika sebuah pin input Arduino tidak terhubung ke VCC (HIGH) atau GND (LOW) secara pasti (misalnya saat tombol tidak ditekan dalam rangkaian sederhana), pin tersebut berada dalam kondisi *floating* (mengambang). Pembacaan `digitalRead()` pada pin ini bisa menjadi acak (kadang HIGH, kadang LOW), membuat program tidak bisa diandalkan.

Solusi: Rangkaian Pull-up atau Pull-down
Untuk mengatasi floating, kita menggunakan resistor untuk "menarik" pin input ke level tegangan yang pasti saat tombol tidak aktif.

1. Rangkaian Pull-down:
   a. Tombol menghubungkan pin input ke VCC (5V) saat ditekan.
   b. Resistor (misal 10kΩ) menghubungkan pin input ke GND.
   c. Saat tidak ditekan: Pin ditarik ke LOW oleh resistor.
   d. Saat ditekan: Pin terhubung ke HIGH.
   e. Pembacaan: LOW (tidak ditekan), HIGH (ditekan).
   f. `pinMode(pinTombol, INPUT);`

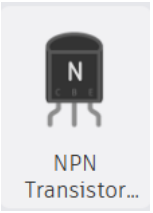


Gambar 8 Rangkaian Pull-down

2. Rangkaian Pull-up (Internal): Cara Paling Mudah di Arduino
   a. Arduino memiliki resistor pull-up internal yang bisa diaktifkan via kode.
   b. Tombol menghubungkan pin input ke GND saat ditekan.
   c. Tidak perlu resistor eksternal.
   d. Saat tidak ditekan: Pin ditarik ke HIGH oleh resistor internal.
   e. Saat ditekan: Pin terhubung ke LOW.
   f. Pembacaan: HIGH (tidak ditekan), LOW (ditekan).
   g. `pinMode(pinTombol, INPUT_PULLUP);`

Karena kemudahannya, menggunakan resistor pull-up internal (`INPUT_PULLUP`) sangat direkomendasikan untuk menghubungkan tombol ke Arduino.

Transistor: Saklar Elektronik Bertenaga



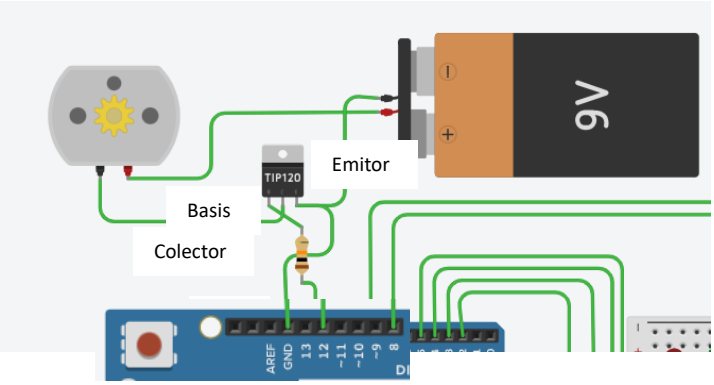
Pin output Arduino hanya dapat menyediakan arus yang terbatas (sekitar 20mA-40mA per pin). Ini cukup untuk menyalakan LED, tetapi tidak cukup untuk mengendalikan perangkat yang membutuhkan arus lebih besar, seperti motor DC, relay, atau strip LED yang panjang.

Di sinilah transistor berperan. Transistor adalah komponen semikonduktor yang dapat berfungsi sebagai saklar elektronik atau penguat.

Fungsi sebagai Saklar:
Dalam konteks Arduino, kita sering menggunakan transistor (khususnya tipe BJT NPN seperti 2N2222 atau BC547, atau MOSFET) sebagai saklar. Pin output Arduino memberikan sinyal kontrol (arus kecil ke Basis BJT, atau tegangan ke Gerbang MOSFET) untuk mengontrol aliran arus yang jauh lebih besar melalui jalur Kolektor-Emitor (BJT) atau Drain-Source (MOSFET) yang terhubung ke beban (misalnya motor).

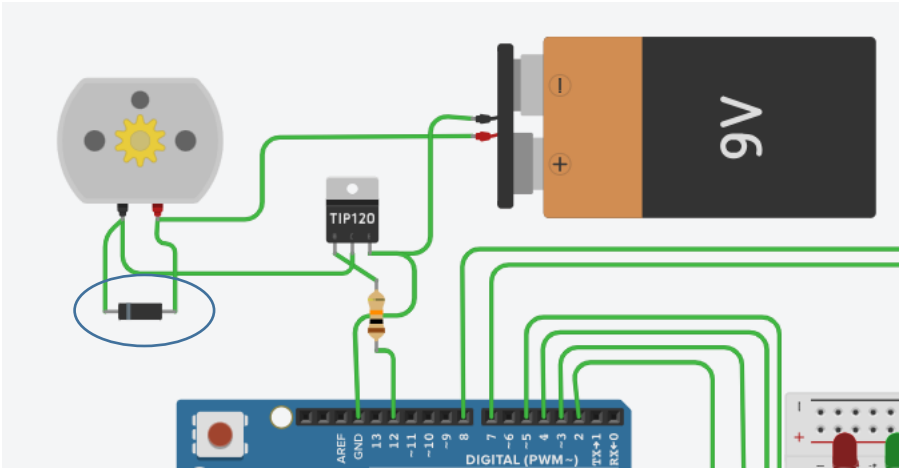
Cara Kerja Sederhana (NPN BJT sebagai Saklar):

1. Beban (misal motor) terhubung antara sumber tegangan positif (yang mungkin lebih tinggi dari 5V) dan Kolektor transistor.
2. Emitor transistor terhubung ke Ground (GND).
3. Basis transistor terhubung ke pin output Arduino melalui sebuah resistor pembatas arus basis (misal 1kΩ).
4. Ketika pin Arduino mengirim sinyal HIGH, arus kecil mengalir ke Basis, "membuka" jalur antara Kolektor dan Emitor, sehingga arus besar dapat mengalir melalui beban (motor berputar).
5. Ketika pin Arduino mengirim sinyal LOW, tidak ada arus Basis, jalur Kolektor-Emitor "tertutup", dan arus ke beban terputus (motor berhenti).



Gambar 9 Rangkaian Transistor mengendalikan Motor

***Catatan Penting:** Saat menggunakan transistor untuk mengendalikan beban induktif seperti motor atau relay, seringkali diperlukan dioda *flyback* (dipasang paralel terbalik dengan beban) untuk melindungi transistor dari lonjakan tegangan saat beban dimatikan.*



Gambar 10 Rangkaian Transistor mengendalikan motor dilengkapi diode flyback

Meskipun detail cara kerja transistor bisa kompleks, pemahaman dasarnya sebagai saklar yang dikontrol Arduino sudah cukup untuk banyak proyek pemula.
Dengan pemahaman tentang komponen-komponen dasar ini, Anda siap untuk mulai belajar bagaimana memprogram Arduino untuk berinteraksi dengan mereka di bab selanjutnya.

Referensi:

- https://www.kompas.com/skola/read/2022/06/21/103000269/fungsi-resistor-pada-rangkaian-elektronika
- https://www.arduino.biz.id/2021/01/pengertian-transistor-jenis-fungsi-dan.html
- https://medium.com/@arisadityanugraha/menggunakan-push-button-pada-arduino-a149337288ff

Bab 3: Mengenal Bahasa C & Algoritma Dasar

Setelah memahami perangkat keras Arduino dan dasar-dasar elektronika, kini saatnya kita menyelami inti dari pengendalian Arduino: pemrograman. Arduino diprogram menggunakan bahasa yang didasarkan pada C/C++, sebuah bahasa pemrograman yang kuat dan banyak digunakan. Bab ini akan memperkenalkan Anda pada struktur dasar program Arduino, cara menggunakan variabel, operator, fungsi-fungsi penting untuk mengontrol pin, konsep algoritma dasar (urutan, percabangan, perulangan), cara berkomunikasi dengan komputer melalui port serial, dan teknik penjadwalan non-blocking menggunakan `millis()`.

Struktur Dasar Program Arduino

Setiap program Arduino, atau yang biasa disebut *sketch*, memiliki struktur dasar yang harus ada. Minimal, sebuah sketch terdiri dari dua fungsi utama: `setup()` dan `loop()`. Selain itu, biasanya ada bagian untuk deklarasi variabel global.

1. Deklarasi Variabel Global:
   a. Variabel yang dideklarasikan di luar semua fungsi (biasanya di bagian paling atas sketch).
   b. Dapat dikenali dan digunakan oleh semua fungsi dalam sketch (`setup()`, `loop()`, dan fungsi buatan sendiri).
   Contoh: `const int pinLed = 13;`

3. Fungsi `setup()`:
   a. Wajib ada.
   b. Dijalankan hanya sekali saat Arduino pertama kali dinyalakan atau setelah tombol reset ditekan.
   c. Digunakan untuk melakukan inisialisasi atau pengaturan awal yang diperlukan sebelum program utama berjalan. Contohnya:
   d. Mengatur mode pin menggunakan **`pinMode()`**.
   e. Memulai komunikasi serial dengan **`Serial.begin()`**.
   f. Menginisialisasi library eksternal.

```cpp
void setup() {
    // Kode inisialisasi di sini
    pinMode(pinLed, OUTPUT); // Contoh: mengatur pin LED sebagai output
    Serial.begin(9600);      // Contoh: memulai komunikasi serial
}
```

4. Fungsi `loop()`:
   a. Wajib ada.
   b. Dijalankan secara terus-menerus dan berulang-ulang setelah `setup()` selesai.
   c. Ini adalah inti program tempat logika utama Anda berjalan. Arduino akan mengeksekusi kode di dalam `loop()` dari atas ke

bawah, lalu kembali lagi ke atas, dan mengulanginya tanpa henti.

```cpp
  void loop() {
          // Kode utama program di sini, akan diulang terus-menerus
          digitalWrite(pinLed, HIGH); // Contoh: menyalakan LED
          delay(500);               // Contoh: jeda 500ms
          digitalWrite(pinLed, LOW);  // Contoh: mematikan LED
          delay(500);               // Contoh: jeda 500ms
  }
```

Contoh menyalakan Lampu berdasar Tombol

```
// C++ code
//
const int LedR=2;
const int LedG=3;
const int LedO=4;
const int LedW=5;
const int btn1=7;
const int btn2=8;

void setup()
{
 for (int a=LedR ; a<=LedW ; a++)
    pinMode(a, OUTPUT);
 for (int a=btn1 ; a<=btn2 ; a++)
    pinMode(a, INPUT_PULLUP);
 for (int a=LedR ; a<=LedW ; a++)
    digitalWrite(a, HIGH);
 initLed();
}

void initLed()
{
  for (int a=LedR ; a<=LedW ; a++)
  {digitalWrite(a, LOW);
   delay(500);
  }
  for (int a=LedW ; a>=LedR ; a--)
  {digitalWrite(a, HIGH);
   delay(500);
  }


}
void loop()
{
  int readbtn1=digitalRead(btn1);
  int readbtn2=digitalRead(btn2);
  if (readbtn1==LOW)
```

```
  {
     for (int a=LedR ; a<=LedW ; a++)
      {digitalWrite(a, LOW);
       delay(500);
       digitalWrite(a,HIGH);
      }
  }
   if (readbtn2==LOW)
  {
     for (int a=LedW ; a>=LedR ; a--)
      {digitalWrite(a, LOW);
       delay(500);
       digitalWrite(a,HIGH);
      }
  }

}
```

Variabel dan Tipe Data

Variabel adalah wadah untuk menyimpan data yang dapat berubah selama program berjalan. Setiap variabel harus memiliki tipe data yang menentukan jenis nilai yang dapat disimpannya.

Tipe Data Umum di Arduino:

- `int`: Menyimpan bilangan bulat (tanpa desimal), positif atau negatif. Contoh: `int jumlahSiswa = 25;`
- `float`: Menyimpan bilangan desimal (pecahan). Contoh: `float suhu = 27.5;`
- `boolean`: Menyimpan nilai logika `true` (benar) atau `false` (salah). Contoh: `boolean lampuNyala = true;`
- `char`: Menyimpan satu karakter tunggal (diapit tanda kutip tunggal). Contoh: `char pilihan = 'A';`
- `String`: Menyimpan urutan karakter (teks) (diapit tanda kutip ganda). Lebih fleksibel tetapi menggunakan lebih banyak memori daripada array char. Contoh: `String pesan = "Halo Arduino!";`
- `byte`: Menyimpan bilangan bulat positif kecil (0 hingga 255).
- `long`: Menyimpan bilangan bulat dengan jangkauan yang lebih besar dari `int`.
- `unsigned long`: Menyimpan bilangan bulat positif dengan jangkauan sangat besar. Tipe data ini dikembalikan oleh fungsi `millis()`.

Deklarasi Variabel:

`tipeData namaVariabel;` atau `tipeData namaVariabel = nilaiAwal;`
Contoh: `int counter = 0;`

Variabel Lokal vs Global:

- Global: Dideklarasikan di luar fungsi, bisa diakses dari mana saja.
- Lokal: Dideklarasikan di dalam fungsi, hanya bisa diakses di dalam fungsi tersebut.

Operator

Operator digunakan untuk melakukan operasi pada variabel dan nilai.

- Operator Aritmatika: `+` (penjumlahan), `-` (pengurangan), `*` (perkalian), `/` (pembagian), `%` (modulo/sisa bagi).
- Operator Perbandingan: `==` (sama dengan), `!=` (tidak sama dengan), `<` (kurang dari), `>` (lebih dari), `<=` (kurang dari atau sama dengan), `>=` (lebih dari atau sama dengan). Hasilnya adalah `boolean` (`true` atau `false`).
- Operator Logika: `&&` (DAN - `true` jika kedua sisi `true`), `||` (ATAU - `true` jika salah satu sisi `true`), `!` (TIDAK - membalikkan nilai `boolean`).
- Operator Penugasan: `=` (mengisi nilai), `+=`, `-=`, `*=`, `/=` (operasi sekaligus penugasan).
- Operator Increment/Decrement: `++` (tambah 1), `--` (kurang 1).

Fungsi Dasar Input/Output (I/O)

Ini adalah fungsi-fungsi inti untuk berinteraksi dengan pin-pin Arduino.

- **`pinMode(pin, mode)`:**
  - Mengatur fungsi sebuah pin digital.
  - `pin`: Nomor pin yang ingin diatur.
  - `mode`: Bisa berupa `INPUT` (untuk membaca sinyal), `OUTPUT` (untuk mengirim sinyal), atau `INPUT_PULLUP` (input dengan resistor pull-up internal aktif).
  - Biasanya dipanggil di dalam `setup()`.
  - Contoh: `pinMode(13, OUTPUT);`

- **`digitalWrite(pin, value)`:**
  - Menulis nilai digital (HIGH atau LOW) ke pin yang sudah diatur sebagai `OUTPUT`.
  - `pin`: Nomor pin.
  - `value`: `HIGH` (biasanya 5V atau 3.3V) atau `LOW` (0V/GND).
  - Contoh: `digitalWrite(13, HIGH);` // Menyalakan LED di pin 13

- **`digitalRead(pin)`:**
  - Membaca nilai digital (HIGH atau LOW) dari pin yang sudah diatur sebagai `INPUT` atau `INPUT_PULLUP`.
  - `pin`: Nomor pin.
  - Mengembalikan `HIGH` atau `LOW`.
  - Contoh: `int statusTombol = digitalRead(2);`

- **`analogRead(pin)`:**
  - Membaca nilai analog dari pin analog input (A0-A5 pada UNO).
  - `pin`: Nomor pin analog (misal: `A0`).
  - Mengembalikan nilai integer antara 0 (untuk 0V) hingga 1023 (untuk 5V atau tegangan referensi AREF).
  - Contoh: `int nilaiSensor = analogRead(A0);`

- **`analogWrite(pin, value)`:**
  - Menulis nilai analog "semu" (PWM - Pulse Width Modulation) ke pin digital yang mendukung PWM (ditandai '~' pada UNO, misal pin 3, 5, 6, 9, 10, 11).
  - `pin`: Nomor pin PWM.

- `value`: Nilai antara 0 (selalu OFF) hingga 255 (selalu ON). Nilai di antaranya menghasilkan sinyal PWM dengan *duty cycle* bervariasi.
- Berguna untuk mengontrol kecerahan LED atau kecepatan motor DC.
- Contoh: `analogWrite(9, 128);` // Mengatur output PWM pin 9 ke 50% duty cycle

Algoritma Dasar

Algoritma adalah urutan langkah logis untuk menyelesaikan masalah. Dalam pemrograman, kita menggunakan struktur kontrol untuk mengimplementasikan algoritma.

1. Sekuensial (Sequence): Instruksi dieksekusi satu per satu, baris demi baris. Ini adalah alur dasar di dalam `setup()` dan `loop()`.

2. Percabangan (Selection): Memilih jalur eksekusi berdasarkan kondisi.
   a. `if (kondisi) { ... }`: Jalankan blok kode jika kondisi `true`.
   b. `if (kondisi) { ... } else { ... }`: Jalankan blok pertama jika `true`, blok kedua jika `false`.
   c. `if (kondisi1) { ... } else if (kondisi2) { ... } else { ... }`: Mengecek beberapa kondisi berurutan.
   d. `switch (variabel) { case nilai1: ... break; case nilai2: ... break; default: ... }`: Membandingkan variabel dengan beberapa nilai konstan.

```cpp
int suhu = analogRead(A0);
if (suhu > 500) {
  Serial.println("Panas!");
} else {
  Serial.println("Normal");
}
```

3. Perulangan (Looping): Menjalankan blok kode berulang kali.
   a. `for (inisialisasi; kondisi; iterasi) { ... }`: Biasanya untuk perulangan dengan jumlah iterasi yang diketahui.
   b. `while (kondisi) { ... }`: Mengecek kondisi sebelum iterasi, berjalan selama kondisi `true`.
   c. `do { ... } while (kondisi);`: Menjalankan kode minimal sekali, lalu cek kondisi.

```cpp
for (int i = 0; i < 5; i++) {
  Serial.print("Iterasi ke: ");
  Serial.println(i);
}
```

Fungsi/Sub-program

Anda dapat membuat fungsi sendiri untuk mengelompokkan kode yang melakukan tugas spesifik. Ini membuat program lebih terstruktur, mudah dibaca, dan kodenya bisa digunakan kembali.

```cpp
// Deklarasi fungsi (bisa di atas atau di bawah setup/loop)
void kedipkanLed(int pin, int durasi) {
  digitalWrite(pin, HIGH);
  delay(durasi);
  digitalWrite(pin, LOW);
  delay(durasi);
}

void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  kedipkanLed(13, 250); // Memanggil fungsi buatan sendiri}
```

Komunikasi Serial

Cara Arduino berkomunikasi dengan komputer (melalui Serial Monitor) atau perangkat lain.

- `Serial.begin(baudRate)`: Memulai komunikasi serial (biasanya di `setup()`). `baudRate` umum adalah 9600.
- `Serial.print(data)`: Mengirim data sebagai teks ke port serial tanpa baris baru.
- `Serial.println(data)`: Mengirim data sebagai teks, diikuti karakter baris baru.
- `Serial.available()`: Mengecek apakah ada data yang masuk dan tersedia untuk dibaca. Mengembalikan jumlah byte yang tersedia.
- `Serial.read()`: Membaca satu byte data yang masuk dari buffer serial. Mengembalikan nilai byte (0-255) atau -1 jika tidak ada data.

```cpp
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available() > 0) {
    char dataMasuk = Serial.read(); // Baca satu karakter
    Serial.print("Anda mengirim: ");
    Serial.println(dataMasuk);
  }
}
```

Konsep Pewaktu `millis()` (Non-Blocking)

Fungsi `delay()` sangat mudah digunakan tetapi memiliki kelemahan besar: ia memblokir atau menghentikan total eksekusi program selama durasi yang ditentukan. Saat `delay()` aktif, Arduino tidak bisa melakukan hal lain.

Fungsi `millis()` adalah solusi untuk membuat jeda atau penjadwalan tanpa memblokir. Ia mengembalikan jumlah milidetik sejak Arduino mulai berjalan (sebagai `unsigned long`).

Pola Umum Penggunaan `millis()`:

1. Simpan waktu terakhir suatu aksi dilakukan (`previousMillis`).
2. Di dalam `loop()`, dapatkan waktu saat ini (`currentMillis = millis();`).
3. Periksa apakah selisih waktu saat ini dan waktu terakhir sudah mencapai interval yang diinginkan (`if (currentMillis - previousMillis >= interval)`).
4. Jika ya, lakukan aksi dan segera update `previousMillis = currentMillis;`.
5. Kode lain di dalam `loop()` tetap bisa berjalan tanpa terganggu.

```cpp
unsigned long previousMillis = 0;
const long interval = 1000; // Interval 1 detik
int ledState = LOW;

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  unsigned long currentMillis = millis();

  // Bagian penjadwalan LED
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis; // Update waktu terakhir
    // Balikkan status LED
    ledState = !ledState;
    digitalWrite(LED_BUILTIN, ledState);
  }

  // Arduino bisa melakukan tugas lain di sini secara bersamaan
  // Misalnya, membaca sensor atau tombol
}
```

Pemahaman tentang bahasa C dasar, struktur algoritma, fungsi I/O, komunikasi serial, dan `millis()` ini akan menjadi fondasi kuat Anda untuk membangun berbagai proyek Arduino yang lebih kompleks.

Referensi:

- https://mikroavr.com/struktur-dasar-program-arduino/
- https://mamikos.com/info/struktur-dasar-algoritma-pljr/

- https://www.arduino.cc/reference/tr/language/functions/communication/serial/print
- https://docs.arduino.cc/language-reference/en/functions/communication/serial/read/
- https://www.norwegiancreations.com/2017/09/arduino-tutorial-using-millis-instead-of-delay/

Bab 4: Implementasi Dasar: Kendali LED & Tombol

Setelah mempelajari dasar-dasar elektronika dan pemrograman Arduino, kini saatnya menggabungkan pengetahuan tersebut untuk membuat interaksi sederhana. Bab ini akan memandu Anda melalui implementasi paling dasar: mengontrol output berupa LED dan membaca input dari tombol tekan (push button). Kita akan mulai dengan program "Hello, World!" versi hardware, yaitu membuat LED berkedip, lalu belajar membaca input tombol, dan mengombinasikannya untuk mengontrol LED dengan tombol.

Blink: Membuat LED Berkedip (Menggunakan `delay()`)

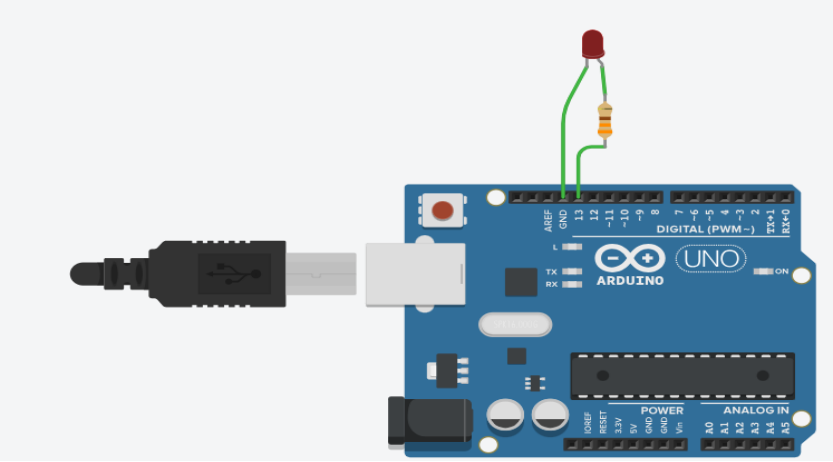
Ini adalah program pertama yang biasanya dicoba oleh pemula Arduino. Tujuannya sederhana: membuat LED menyala dan mati secara bergantian.

Komponen yang Dibutuhkan:
- Arduino Board (UNO, Nano, dll.)
- Kabel USB
- (Opsional, jika tidak menggunakan LED bawaan) 1x LED, 1x Resistor 220Ω
- 1kΩ, Kabel Jumper, Breadboard.

Rangkaian:
Cara termudah adalah menggunakan LED bawaan (*built-in LED*) yang ada di board Arduino. LED ini biasanya terhubung ke pin digital 13 (pada UNO/Nano). Arduino menyediakan konstanta `LED_BUILTIN` yang merujuk ke pin ini, sehingga kode lebih portabel.

Jika menggunakan LED eksternal:

1. Hubungkan kaki pendek LED (Katoda) ke GND Arduino.
2. Hubungkan kaki panjang LED (Anoda) ke salah satu ujung resistor (misal 330Ω).
3. Hubungkan ujung resistor yang lain ke pin digital `LED_BUILTIN` (atau pin 13 pada UNO).



Gambar 11 Wirring Led Berkedip

Kode Program:
```cpp
// Program Blink Dasar

void setup() {
  // Mengatur pin LED_BUILTIN sebagai output
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // Menyalakan LED
  delay(1000);                     // Jeda selama 1000 milidetik (1 detik)
  digitalWrite(LED_BUILTIN, LOW);  // Mematikan LED
  delay(1000);                     // Jeda selama 1000 milidetik (1 detik)
}
```

Cara Kerja:

- `setup()`: Mengatur pin `LED_BUILTIN` sebagai `OUTPUT`.
  `loop()`:
- `digitalWrite(LED_BUILTIN, HIGH);`: Memberikan tegangan ke pin, LED menyala.
- `delay(1000);`: Program berhenti total selama 1 detik.
- `digitalWrite(LED_BUILTIN, LOW);`: Menghentikan tegangan ke pin, LED mati.
- `delay(1000);`: Program berhenti total lagi selama 1 detik.
- Proses di dalam `loop()` ini terus berulang.

Blink Without Delay: Mengedipkan LED Menggunakan `millis()`

Seperti yang dibahas di Bab 3, `delay()` memblokir program. Jika kita ingin Arduino melakukan hal lain sambil LED berkedip (misalnya membaca sensor), kita perlu menggunakan `millis()` menggunakan Wirring gambar 11 .

Kode Program (Blink Without Delay):
```cpp
// Program Blink Tanpa Delay

const int ledPin = LED_BUILTIN; // Pin LED
int ledState = LOW;             // Menyimpan status LED saat ini (LOW atau HIGH)

unsigned long previousMillis = 0; // Menyimpan waktu terakhir LED diubah statusnya
const long interval = 1000;     // Interval kedip (1000 ms = 1 detik)

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
```

```
  // Dapatkan waktu saat ini
  unsigned long currentMillis = millis();

  // Cek apakah sudah waktunya mengubah status LED
  if (currentMillis - previousMillis >= interval) {
    // Simpan waktu saat ini sebagai waktu terakhir perubahan
    previousMillis = currentMillis;

    // Ubah status LED: jika LOW jadi HIGH, jika HIGH jadi LOW
    if (ledState == LOW) {
      ledState = HIGH;
    } else {
      ledState = LOW;
    }

    // Terapkan status baru ke LED
    digitalWrite(ledPin, ledState);
  }

  // Program bisa melakukan hal lain di sini tanpa terganggu oleh
kedipan LED
  // Misalnya: Serial.println("Arduino sedang bekerja...");
}
```

Cara Kerja:

- Program tidak berhenti. Di setiap iterasi `loop()`, ia memeriksa apakah sudah 1 detik berlalu sejak status LED terakhir diubah.
- Jika sudah, ia mengubah status LED (`HIGH` jadi `LOW` atau sebaliknya) dan mencatat waktu perubahan tersebut.
- Ini memungkinkan Arduino tetap responsif terhadap tugas lain.

Membaca Tombol (Push Button)

Sekarang kita akan belajar membaca input dari pengguna melalui tombol.

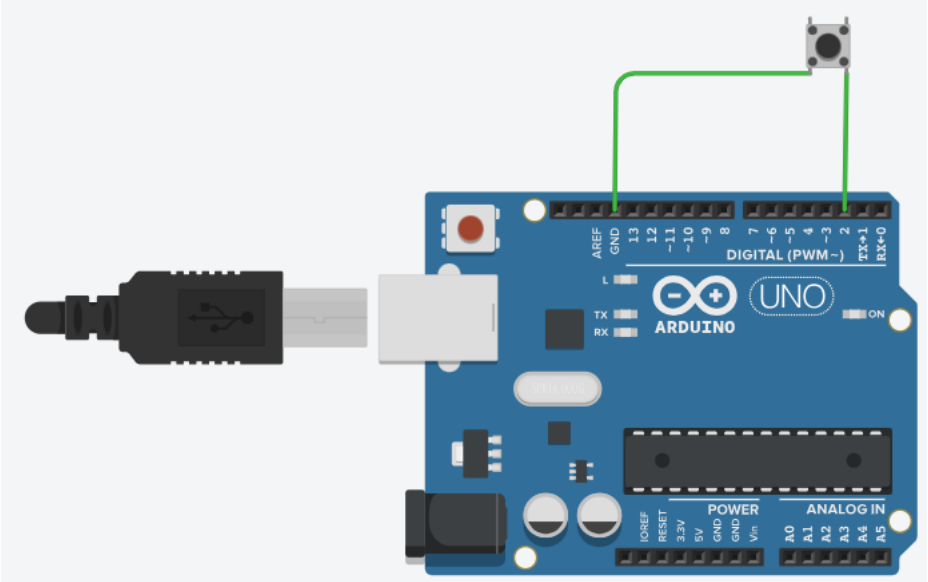
Komponen yang Dibutuhkan:

- Arduino Board
- Kabel USB
- 1x Push Button
- (Opsional, jika tidak menggunakan pull-up internal) 1x Resistor 10kΩ
- Kabel Jumper
- Breadboard

Rangkaian (Menggunakan Internal Pull-up - Direkomendasikan):

1. Pasang tombol di breadboard.
2. Hubungkan salah satu kaki tombol ke pin digital Arduino (misal pin 2).
3. Hubungkan kaki tombol *di seberangnya* (yang terhubung secara internal saat ditekan) ke GND Arduino.
4. Tidak perlu resistor eksternal.

Kode Program (Membaca Tombol dengan Internal Pull-up):



Gambar 12 Rangkaian Push Button

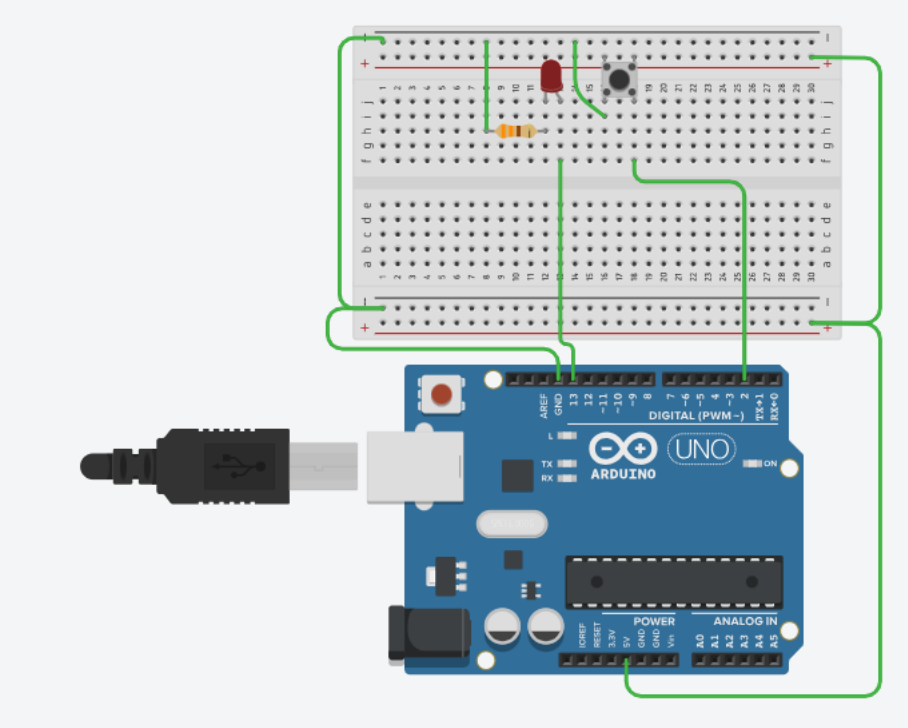
```cpp
// Program Membaca Tombol (Internal Pull-up)

const int buttonPin = 2; // Pin tombol

void setup() {
  Serial.begin(9600); // Mulai komunikasi serial untuk melihat hasil
  // Atur pin tombol sebagai INPUT dengan resistor pull-up internal
aktif
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  // Baca status tombol
  int buttonState = digitalRead(buttonPin);

  // Tampilkan status ke Serial Monitor
  // (LOW jika ditekan, HIGH jika tidak ditekan karena pull-up)
  Serial.println(buttonState);

  delay(50); // Beri jeda sedikit untuk stabilitas pembacaan
}
```

Cara Kerja:

- `pinMode(buttonPin, INPUT_PULLUP);`: Mengatur pin 2 sebagai input dan mengaktifkan resistor pull-up internal. Ini membuat pin 2 membaca HIGH saat tombol tidak ditekan.
- `digitalRead(buttonPin);`: Membaca status pin 2.
- Saat tombol tidak ditekan, pin 2 ditarik ke HIGH oleh resistor internal, `digitalRead()` mengembalikan `HIGH` (1).
- Saat tombol ditekan, pin 2 terhubung ke GND, `digitalRead()` mengembalikan `LOW` (0).
- Hasilnya (1 atau 0) dikirim ke Serial Monitor.

Mengontrol LED dengan Tombol

Mari gabungkan kedua contoh sebelumnya: menyalakan LED hanya saat tombol ditekan.



Gambar 13 Gabungan kedua Contoh diatas LED dan Push Button

Rangkaian: Sama seperti rangkaian membaca tombol (internal pull-up), ditambah LED (bisa LED bawaan `LED_BUILTIN` atau LED eksternal dengan resistor di pin 13).

Kode Program:

```cpp
// Program Kontrol LED dengan Tombol (Internal Pull-up)

const int buttonPin = 2;     // Pin tombol
const int ledPin = LED_BUILTIN; // Pin LED

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  // Baca status tombol
  int buttonState = digitalRead(buttonPin);

  // Cek apakah tombol ditekan (LOW karena pull-up)
  if (buttonState == LOW) {
```

```
    // Jika ditekan, nyalakan LED
    digitalWrite(ledPin, HIGH);
  } else {
    // Jika tidak ditekan, matikan LED
    digitalWrite(ledPin, LOW);
  }
}
```

Cara Kerja:

- Program terus menerus membaca status tombol di dalam `loop()`.
- Jika `digitalRead()` mengembalikan `LOW` (artinya tombol ditekan), maka `digitalWrite()` digunakan untuk menyalakan LED (`HIGH`).
- Jika `digitalRead()` mengembalikan `HIGH` (tombol tidak ditekan), maka LED dimatikan (`LOW`).

Kasus Lain (Contoh Pengembangan)

1. Toggle LED: Membuat status LED berubah (ON jadi OFF, OFF jadi ON) setiap kali tombol ditekan. Ini memerlukan logika tambahan untuk mendeteksi *perubahan* status tombol (dari tidak ditekan menjadi ditekan) dan variabel untuk menyimpan status LED terakhir.

```cpp
// Contoh Logika Toggle (perlu disempurnakan dengan debouncing)
int ledState = LOW;
int lastButtonState = HIGH; // Asumsi tombol tidak ditekan awalnya
unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 50;

void loop() {
  int reading = digitalRead(buttonPin);
  unsigned long currentMillis = millis();

  if (reading != lastButtonState) {
    lastDebounceTime = currentMillis;
  }

  if ((currentMillis - lastDebounceTime) > debounceDelay) {
    if (reading != buttonState) {
      buttonState = reading;
      if (buttonState == LOW) { // Tombol baru saja ditekan
        ledState = !ledState; // Balikkan status LED
        digitalWrite(ledPin, ledState);
      }
    }
  }
  lastButtonState = reading;
}
```

Catatan: Kode toggle di atas menyertakan konsep dasar "debouncing" menggunakan `millis()` untuk mengatasi pembacaan ganda saat tombol ditekan/dilepas.

2. Mengontrol Beberapa LED: Anda bisa menggunakan beberapa tombol untuk mengontrol beberapa LED secara independen, atau satu tombol untuk mengontrol pola nyala beberapa LED.

Eksperimen dengan contoh-contoh ini akan memperkuat pemahaman Anda tentang input dan output digital, serta logika pemrograman dasar di Arduino.

Referensi:

- https://docs.arduino.cc/built-in-examples/basics/Blink/
- https://docs.arduino.cc/built-in-examples/digital/Button/
- https://www.norwegiancreations.com/2017/09/arduino-tutorial-using-millis-instead-of-delay/

Bab 5: Membaca Sensor & Menampilkan ke LCD

Setelah mampu mengontrol output dasar seperti LED dan membaca input sederhana dari tombol, langkah selanjutnya adalah membuat Arduino berinteraksi lebih jauh dengan lingkungannya menggunakan sensor dan menampilkan informasi yang lebih kompleks menggunakan LCD (Liquid Crystal Display).

Bab ini akan memperkenalkan Anda pada beberapa sensor populer yang mudah digunakan oleh pemula: sensor suhu dan kelembaban (DHT11), sensor cahaya (LDR/Photoresistor), dan sensor jarak ultrasonik (HC-SR04). Kita juga akan belajar cara menampilkan data dari sensor-sensor ini (atau data lainnya) ke layar LCD karakter 16x2 menggunakan antarmuka I2C yang hemat pin.

Sensor: Mata dan Telinga Arduino

Sensor adalah perangkat yang mendeteksi perubahan atau kejadian di lingkungan fisik dan mengubahnya menjadi sinyal listrik yang dapat dibaca oleh mikrokontroler seperti Arduino. Ada berbagai jenis sensor untuk mengukur besaran fisik yang berbeda.

1. Sensor Suhu dan Kelembaban (DHT11)

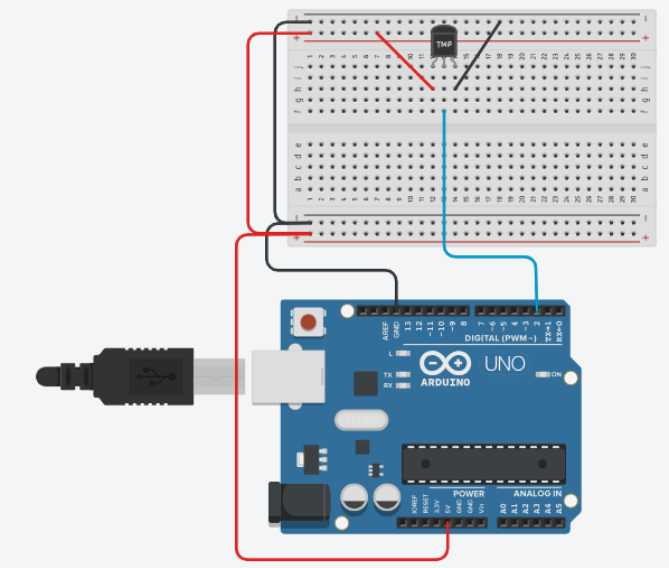
DHT11 adalah sensor digital berbiaya rendah yang populer untuk mengukur suhu udara dan kelembaban relatif (RH).

- Cara Kerja: Menggunakan sensor kelembaban kapasitif dan termistor untuk mengukur kondisi udara sekitar. Sebuah chip di dalamnya memproses pembacaan dan mengirimkan data digital melalui satu pin data.
- Spesifikasi Umum:
  1. Rentang Suhu: 0-50 °C (Akurasi ±2°C)
  2. Rentang Kelembaban: 20-90% RH (Akurasi ±5%)
  3. Tegangan: 3-5.5V
- Jenis: Ada yang 4 pin (membutuhkan resistor pull-up eksternal 10kΩ antara VCC dan Data) dan modul 3 pin (biasanya sudah termasuk resistor pull-up).
- Koneksi (Modul 3 Pin):
  1. VCC/+: ke 5V Arduino
  2. GND/-: ke GND Arduino
  3. Data/OUT/S: ke Pin Digital Arduino (misal, pin 2)
- Library: Memerlukan library khusus. "DHT sensor library" dari Adafruit adalah pilihan umum (instal melalui Library Manager, mungkin perlu "Adafruit Unified Sensor" juga).

Contoh Kode Membaca DHT11:

```cpp
include "DHT.h" // Sertakan library

define DHTPIN 2     // Pin data sensor
define DHTTYPE DHT11   // Tipe sensor

DHT dht(DHTPIN, DHTTYPE); // Buat objek sensor
```

```
void setup() {
  Serial.begin(9600);
  dht.begin(); // Inisialisasi sensor
}

void loop() {
  delay(2000); // DHT11 butuh jeda antar pembacaan

  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature(); // Celsius

  // Cek jika pembacaan gagal
  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Gagal membaca DHT11!");
    return;
  }

  Serial.print("Kelembaban: ");
  Serial.print(humidity);
  Serial.print(" %\t"); // \t = tab
  Serial.print("Suhu: ");
  Serial.print(temperature);
  Serial.println(" *C");
}
```



Gambar 14 Rangkaian Sensor panas

2. Sensor Cahaya (LDR/Photoresistor)

LDR (Light-Dependent Resistor) adalah resistor yang nilai hambatannya berubah tergantung intensitas cahaya yang diterimanya: semakin terang, semakin kecil hambatannya.

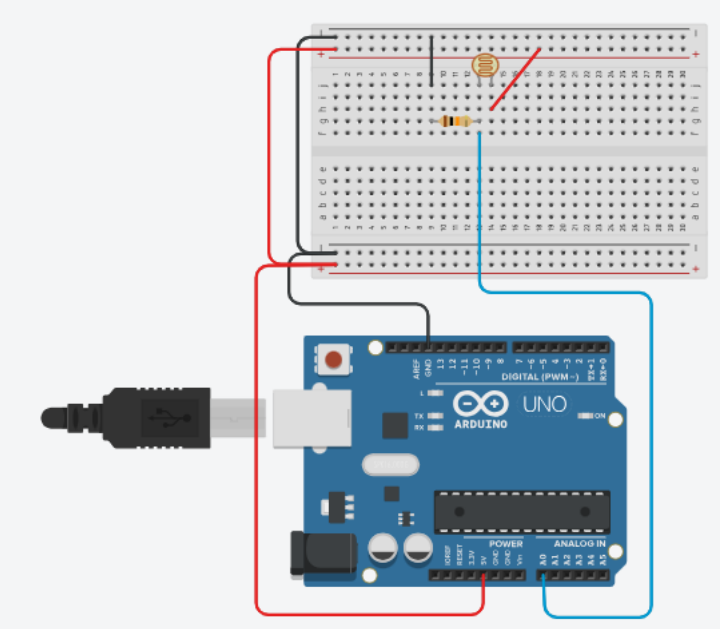
- Cara Kerja dengan Arduino: Karena Arduino membaca tegangan analog, LDR digunakan dalam rangkaian pembagi tegangan dengan resistor tetap (misal 10kΩ). Perubahan resistansi LDR akan menyebabkan

perubahan tegangan pada titik tengah pembagi tegangan, yang kemudian dibaca oleh pin analog Arduino.

- Rangkaian Pembagi Tegangan Umum:
  1. Satu kaki LDR ke 5V.
  2. Kaki LDR lainnya ke Pin Analog (misal A0).
  3. Pin Analog yang sama (A0) dihubungkan ke GND melalui resistor 10kΩ.

Hasil: Tegangan di A0 akan tinggi saat terang, rendah saat gelap.

- Pembacaan: Menggunakan `analogRead(pinAnalog)` yang mengembalikan nilai 0-1023 (proporsional terhadap tegangan 0-5V).



Gambar 15 Rangkaian Sensor Cahaya

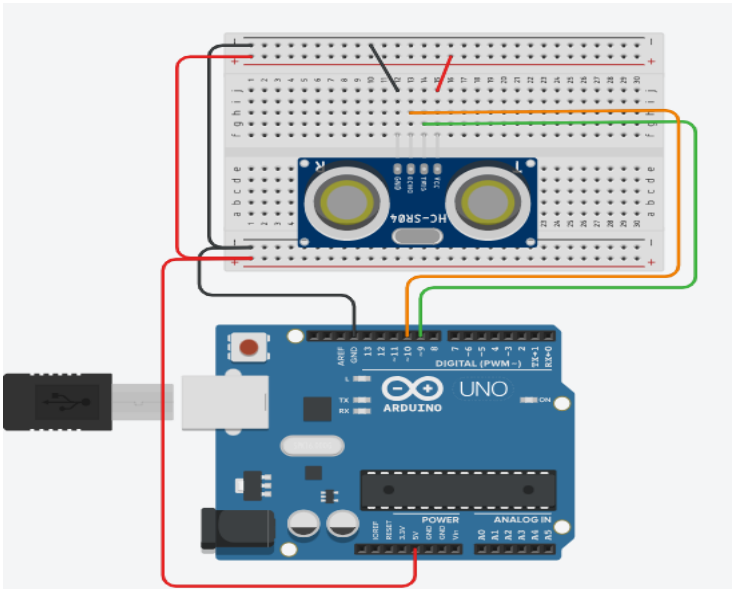
Contoh Kode Membaca LDR:

```cpp
const int ldrPin = A0; // Pin analog LDR

void setup() {
  Serial.begin(9600);
}

void loop() {
  int ldrValue = analogRead(ldrPin);

  Serial.print("Nilai Sensor Cahaya: ");
  Serial.println(ldrValue); // Nilai 0-1023 (tinggi saat terang, rendah saat gelap)

  // Contoh: Nyalakan LED jika gelap (nilai < 300, perlu eksperimen)
  // if (ldrValue < 300) { digitalWrite(LED_BUILTIN, HIGH); }
  // else { digitalWrite(LED_BUILTIN, LOW); }

  delay(100);
}
```

3. Sensor Jarak Ultrasonik (HC-SR04)

Sensor ini mengukur jarak ke objek di depannya menggunakan gelombang suara ultrasonik, mirip seperti sonar kelelawar.

- Cara Kerja:
  1. Arduino mengirim pulsa pendek ke pin `Trig` sensor.
  2. Sensor memancarkan gelombang ultrasonik.
  3. Sensor membuat pin `Echo` menjadi HIGH.
  4. Gelombang memantul dari objek dan kembali ke sensor.
  5. Saat pantulan diterima, sensor membuat pin `Echo` menjadi LOW.
  6. Arduino mengukur durasi pin `Echo` HIGH menggunakan `pulseIn()`.
  7. Jarak dihitung dari durasi tersebut (`Jarak = (Durasi * Kecepatan Suara) / 2`).
- Spesifikasi Umum:
  1. Jarak: 2cm - 400cm
  2. Tegangan: 5V
- Koneksi:
  - VCC: ke 5V Arduino
  - GND: ke GND Arduino
  - Trig: ke Pin Digital Arduino (misal 9, diatur OUTPUT)
  - Echo: ke Pin Digital Arduino (misal 10, diatur INPUT)



Gambar 16 Rangkaian Sensor Ultrasonik

Contoh Kode Membaca HC-SR04:

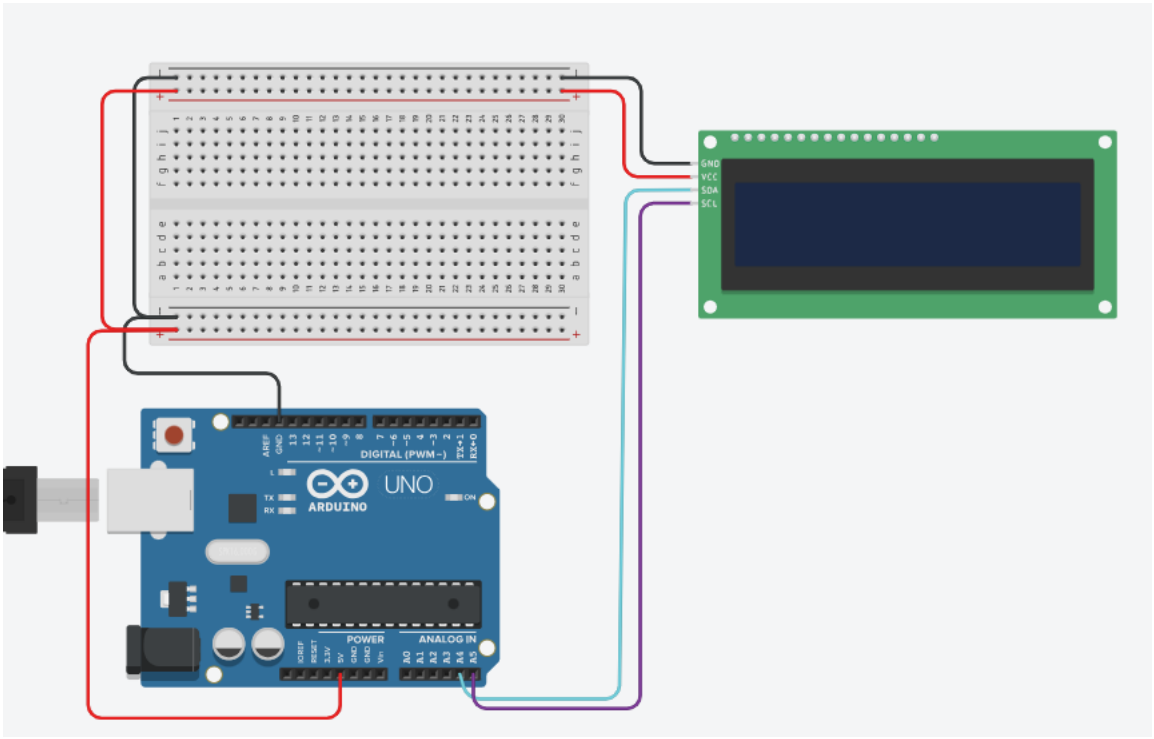
```cpp
const int trigPin = 9;
const int echoPin = 10;

long duration;
int distance; // Jarak dalam cm
```

```
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  // Kirim pulsa trigger 10 mikrosekon
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Baca durasi pulsa echo
  duration = pulseIn(echoPin, HIGH);

  // Hitung jarak (kecepatan suara = 0.034 cm/µs)
  distance = duration * 0.034 / 2;

  // Tampilkan jarak
  Serial.print("Jarak: ");
  Serial.print(distance);
  Serial.println(" cm");

  delay(100);
}
```

Menampilkan Data ke LCD 16x2 (via I2C)

LCD karakter 16x2 (16 kolom, 2 baris) adalah cara populer untuk menampilkan informasi teks atau data sensor tanpa memerlukan komputer.

- Masalah Koneksi Paralel: Menghubungkan LCD standar secara langsung membutuhkan banyak pin Arduino (minimal 6 pin digital).
- Solusi: Modul I2C LCD Adapter: Modul kecil yang dipasang di belakang LCD, memungkinkan kontrol melalui protokol I2C hanya dengan 2 pin data (SDA, SCL) ditambah VCC dan GND.
- Keuntungan I2C: Hemat pin, wiring sederhana, bus I2C bisa dibagi dengan perangkat I2C lain.
- Koneksi ke Arduino UNO:
  - GND Modul: ke GND Arduino
  - VCC Modul: ke 5V Arduino
  - SDA Modul: ke Pin A4 Arduino
  - SCL Modul: ke Pin A5 Arduino
- Alamat I2C: Setiap perangkat I2C punya alamat unik. Modul LCD biasanya beralamat `0x27` atau `0x3F`. Gunakan *I2C Scanner Sketch* (File > Examples > Wire > i2c_scanner) untuk memastikannya jika LCD tidak menampilkan apa-apa.
- Kontras: Putar potensiometer biru di belakang modul I2C untuk mengatur kontras tampilan.
- Library: Memerlukan library `Wire.h` (standar) dan `LiquidCrystal_I2C.h` (instal via Library Manager).



Gambar 17 Rangkaian LCD I2C

Contoh Kode Menampilkan ke LCD I2C:

```cpp
include <Wire.h>
include <LiquidCrystal_I2C.h>

// Ganti 0x27 dengan alamat I2C LCD Anda jika berbeda
LiquidCrystal_I2C lcd(0x27, 16, 2); // Alamat, Kolom, Baris

void setup() {
  lcd.init();      // Inisialisasi LCD
  lcd.backlight(); // Nyalakan backlight

  lcd.setCursor(0, 0); // Kolom 0, Baris 0
  lcd.print("Suhu:");
  lcd.setCursor(0, 1); // Kolom 0, Baris 1
  lcd.print("Lembab:");
}

void loop() {
  // Misalkan kita punya variabel suhu dan kelembaban
  float suhu = 28.5; // Contoh data
  float lembab = 65.2; // Contoh data

  // Tampilkan data ke LCD
  lcd.setCursor(7, 0); // Kolom 7, Baris 0
  lcd.print(suhu);     // Tampilkan nilai suhu
  lcd.print(" C");

  lcd.setCursor(7, 1); // Kolom 7, Baris 1
  lcd.print(lembab);   // Tampilkan nilai kelembaban
```

```
  lcd.print(" %");

  delay(1000); // Jeda sebelum update berikutnya
}
```

Fungsi LCD Penting:
- `lcd.init()` atau `lcd.begin()`: Inisialisasi.
- `lcd.backlight()` / `lcd.noBacklight()`: Kontrol lampu latar.
- `lcd.setCursor(kolom, baris)`: Pindahkan kursor (mulai dari 0).
- `lcd.print(data)`: Tampilkan teks atau nilai variabel.
- `lcd.clear()`: Hapus seluruh tampilan layar.

Dengan kemampuan membaca berbagai jenis sensor dan menampilkan hasilnya ke LCD, Anda dapat mulai membangun proyek-proyek yang lebih informatif dan interaktif, seperti stasiun cuaca mini, pengukur jarak digital, atau sistem otomatisasi sederhana berdasarkan kondisi lingkungan.

Referensi:

- https://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/
- https://www.instructables.com/How-to-use-a-photoresistor-or-photocell-Arduino-Tu/
- https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/
- https://lastminuteengineers.com/i2c-lcd-arduino-tutorial/

BAB 6 Simulasi Arduino dengan Tinkercad

4.1. Pengenalan Tinkercad

Tinkercad adalah platform berbasis web yang disediakan oleh Autodesk untuk desain 3D, simulasi rangkaian elektronik, dan pemrograman mikrokontroler (seperti Arduino). Kelebihan Tinkercad antara lain:

- Gratis dan berbasis online (tidak perlu instalasi).
- Antarmuka ramah pengguna, cocok untuk pelajar dan pemula.
- Mendukung simulasi rangkaian dan upload kode Arduino secara langsung.
- Memiliki koleksi komponen lengkap seperti LED, motor servo, sensor, dan lain-lain.

Alamat situs: https://www.tinkercad.com/

4.2. Langkah-langkah Menggunakan Tinkercad untuk Simulasi Arduino

Langkah 1: Registrasi dan Login

- Buka Tinkercad.
- Daftar akun gratis atau login menggunakan akun Google/Autodesk.

Langkah 2: Membuat Proyek Baru

- Klik "Create New Circuit" pada dashboard.
- Jendela kerja sirkuit akan terbuka.

Langkah 3: Menambahkan Komponen

- Gunakan sidebar di kanan untuk mencari dan menambahkan komponen seperti:
  - Arduino Uno
  - Servo motor
  - Potentiometer
  - Breadboard (jika diperlukan)
  - Resistor, kabel, dll.

Langkah 4: Menghubungkan Komponen

- Gunakan klik-drag untuk menghubungkan kabel antar komponen.
- Gunakan warna kabel berbeda untuk membedakan fungsi (misal: merah = power, hitam = ground, biru = sinyal).

Langkah 5: Menulis Kode Arduino

- Klik tab "Code".
- Gunakan mode "Text" untuk menulis kode C Arduino.
- Klik "Start Simulation" untuk menjalankan simulasi.

4.3. Contoh Kasus: Pengendali Servo Motor Menggunakan Potensiometer

Tujuan

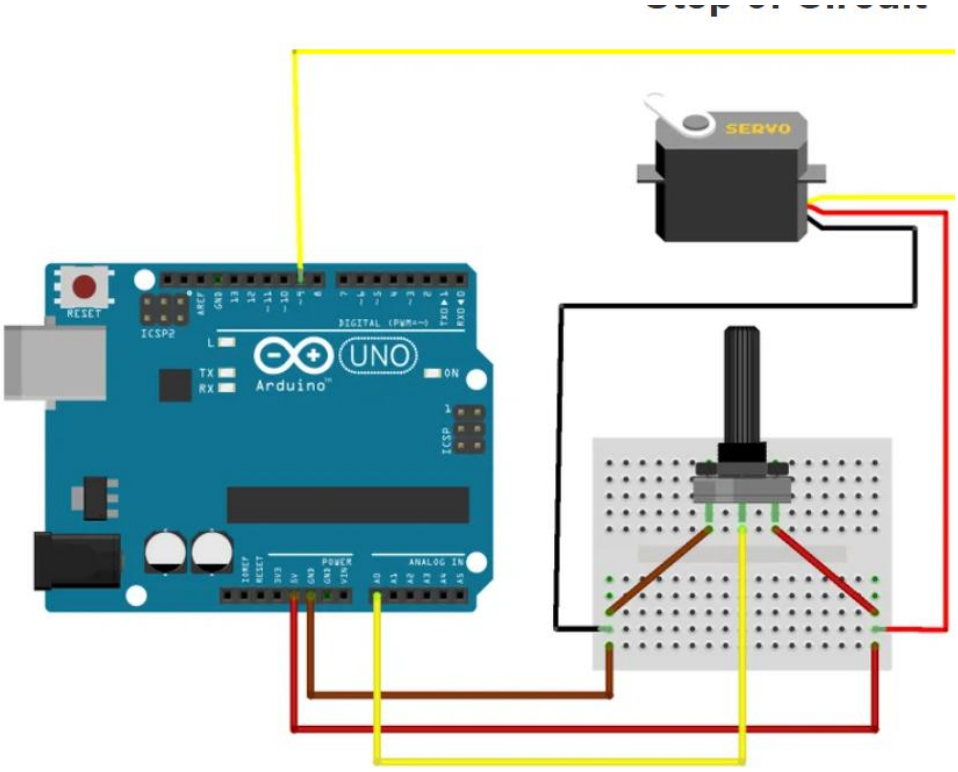
Membuat simulasi Arduino yang menggerakkan **motor servo** berdasarkan nilai **potensiometer** menggunakan Tinkercad.

Komponen yang Digunakan

- 1 x Arduino Uno
- 1 x Servo motor
- 1 x Potensiometer
- Kabel jumper
- (Opsional) Breadboard

Rangkaian

- **Potensiometer:**
  - Kaki tengah → A0 (Arduino)
  - Kaki kiri → 5V
  - Kaki kanan → GND
- **Servo Motor:**
  - Kabel sinyal (kuning/oranye) → Pin 9 (Arduino)
  - VCC (merah) → 5V
  - GND (hitam/coklat) → GND



Gambar 18 Rangkaian Kendali servo dengan potensiometer *(Autodesk, 2025)*

Kode Arduino

cpp

```
#include <Servo.h>

Servo myServo;
int potPin = A0; // pin analog potensiometer
int val;         // variabel pembaca nilai pot

void setup() {
  myServo.attach(9); // hubungkan servo ke pin 9
}

void loop() {
  val = analogRead(potPin);           // baca nilai dari 0-1023
  val = map(val, 0, 1023, 0, 180);    // konversi ke sudut servo (0-180)
  myServo.write(val);                 // gerakkan servo
  delay(15);                          // delay kecil untuk kestabilan
}
```

Langkah Simulasi

1. Hubungkan semua komponen sesuai skema.
2. Salin kode ke editor Arduino di Tinkercad.
3. Klik "Start Simulation".
4. Putar potensiometer dan amati servo bergerak sesuai sudut.

Soal Uji Kompetensi Arduino Dasar

Petunjuk: Pilih satu jawaban yang paling tepat untuk setiap pertanyaan berikut.

Bab 1: Mengenal Board Arduino

a. Apa fungsi utama dari platform Arduino?
b. Sebagai sistem operasi untuk komputer.
c. Sebagai platform open-source untuk membuat prototipe elektronik interaktif.
d. Sebagai bahasa pemrograman khusus untuk robotika.
e. Sebagai jenis baterai untuk perangkat elektronik.

2. Mikrokontroler yang paling umum digunakan pada board Arduino UNO adalah...
a. ATtiny85
b. ESP32
c. ATmega328P
d. Raspberry Pi Pico

3. Pin mana pada Arduino UNO yang digunakan untuk membaca sinyal analog dari sensor?
a. Pin Digital 0-13
b. Pin A0-A5
c. Pin Vin
d. Pin Reset

4. Apa fungsi dari pin GND pada board Arduino?
a. Memberikan tegangan input 5V.
b. Sebagai referensi tegangan ground (0 Volt).
c. Untuk mengunggah program.
d. Untuk komunikasi serial.

5. Software yang digunakan untuk menulis, mengompilasi, dan mengunggah program ke Arduino disebut...
a. Arduino Studio
b. Arduino IDE (Integrated Development Environment)
c. Arduino Compiler
d. Arduino Uploader

Bab 2: Elektronika Dasar Digital

6. Mengapa LED perlu dipasang seri dengan resistor saat dihubungkan ke pin output Arduino?
a. Untuk meningkatkan kecerahan LED.
b. Untuk membatasi arus yang mengalir agar LED tidak rusak.
c. Untuk mengubah warna LED.
d. Untuk membuat LED berkedip.

7. Dalam rangkaian tombol dengan resistor pull-up internal (`INPUT_PULLUP`), apa nilai yang dibaca oleh `digitalRead()` saat tombol ditekan?
a. HIGH

b. LOW
c. Nilai antara 0 dan 1023
d. Tidak menentu (floating)

8. Apa fungsi utama transistor (misal BJT NPN) saat digunakan bersama Arduino untuk mengendalikan motor DC?
a. Sebagai sumber tegangan untuk motor.
b. Sebagai sensor kecepatan motor.
c. Sebagai saklar elektronik untuk mengalirkan arus besar ke motor yang dikontrol oleh sinyal kecil dari Arduino.
d. Sebagai pengatur warna lampu pada motor.

9. Apa yang dimaksud dengan kondisi "floating" pada pin input digital?
a. Pin terhubung langsung ke 5V.
b. Pin terhubung langsung ke GND.
c. Pin tidak terhubung ke level tegangan yang pasti (HIGH atau LOW), sehingga pembacaannya tidak menentu.
d. Pin sedang mengirimkan data.

10. Komponen apa yang digunakan untuk merangkai prototipe elektronik tanpa perlu menyolder?
a. PCB (Printed Circuit Board)
b. Breadboard (Project Board)
c. Multimeter
d. Oscilloscope

Bab 3: Mengenal Bahasa C & Algoritma Dasar

11. Fungsi mana dalam program Arduino yang hanya dijalankan sekali saat startup?
a. `loop()`
b. `main()`
c. `setup()`
d. `delay()`

12. Perintah `digitalWrite(pinLed, HIGH);` digunakan untuk...
a. Membaca status pin `pinLed`.
b. Mengatur `pinLed` sebagai input.
c. Memberikan tegangan HIGH (misal 5V) ke `pinLed`.
d. Memberikan tegangan LOW (0V) ke `pinLed`.

13. Struktur kontrol `if (kondisi) { ... } else { ... }` termasuk dalam jenis algoritma...
a. Sekuensial
b. Perulangan
c. Percabangan
d. Fungsi

14. Apa perbedaan utama antara `delay()` dan menggunakan `millis()` untuk penjadwalan?
a. `delay()` lebih akurat daripada `millis()`.
b. `millis()` hanya bisa digunakan di `setup()`.
c. `delay()` bersifat blocking (menghentikan program), sedangkan `millis()` memungkinkan penjadwalan non-blocking.

d. `millis()` mengembalikan nilai dalam detik, `delay()` dalam milidetik.

15. Fungsi `Serial.begin(9600);` biasanya ditempatkan di dalam...
a. `loop()`
b. `setup()`
c. Deklarasi variabel global
d. Fungsi buatan sendiri

Bab 4: Implementasi Dasar (LED & Button)

16. Kode `pinMode(LED_BUILTIN, OUTPUT);` berfungsi untuk...
a. Menyalakan LED bawaan.
b. Membaca status LED bawaan.
c. Mengatur pin LED bawaan agar dapat mengirim sinyal keluar (mengontrol LED).
d. Mengatur pin LED bawaan agar dapat menerima sinyal masuk.

17. Jika menggunakan `pinMode(buttonPin, INPUT_PULLUP);`, logika `if` untuk mendeteksi tombol ditekan adalah...
a. `if (digitalRead(buttonPin) == HIGH)`
b. `if (digitalRead(buttonPin) == LOW)`
c. `if (analogRead(buttonPin) > 500)`
d. `if (analogRead(buttonPin) < 500)`

18. Apa tujuan dari kode "Blink Without Delay" yang menggunakan `millis()`?
a. Membuat LED berkedip lebih cepat.
b. Membuat LED berkedip tanpa menghentikan eksekusi tugas lain oleh Arduino.
c. Menghemat penggunaan memori Arduino.
d. Menggantikan fungsi `digitalWrite()`.

Bab 5: Membaca Sensor & LCD

19. Sensor DHT11 digunakan untuk mengukur...
a. Jarak dan cahaya
b. Suhu dan kelembaban
c. Tekanan udara dan ketinggian
d. Arus dan tegangan

20. Bagaimana cara membaca nilai dari sensor LDR yang terhubung ke pin A0?
a. `digitalRead(A0);`
b. `digitalWrite(A0, HIGH);`
c. `analogRead(A0);`
d. `analogWrite(A0, 128);`

21. Fungsi `pulseIn(echoPin, HIGH);` pada sensor ultrasonik HC-SR04 digunakan untuk...
a. Mengirim pulsa trigger.
b. Mengukur durasi waktu pin echo berada dalam kondisi HIGH.
c. Menghitung jarak langsung dalam cm.
d. Mengatur frekuensi gelombang ultrasonik.

22. Berapa jumlah pin minimal yang dibutuhkan Arduino untuk berkomunikasi dengan modul LCD 16x2 I2C?
 a. 6 pin (Data, RS, E, VCC, GND)
 b. 4 pin (VCC, GND, SDA, SCL)
 c. 2 pin (SDA, SCL)
 d. 1 pin (Data)

23. Pada Arduino UNO, pin A4 dan A5 secara khusus digunakan untuk komunikasi...
 a. Serial (TX/RX)
 b. SPI (MOSI, MISO, SCK)
 c. I2C (SDA, SCL)
 d. PWM

24. Perintah `lcd.setCursor(0, 1);` pada library `LiquidCrystal_I2C` akan memindahkan kursor ke...
 a. Kolom 1, Baris 0
 b. Kolom 0, Baris 1
 c. Kolom 1, Baris 1
 d. Kolom 0, Baris 0

25. Library apa yang umumnya diperlukan untuk menggunakan sensor DHT11 dengan Arduino?
 a. `Wire.h`
 b. `Servo.h`
 c. `DHT.h` (dan mungkin `Adafruit_Sensor.h`)
 d. `LiquidCrystal_I2C.h`

Kunci Jawaban

1. b
2. c
3. b
4. b
5. b
6. b
7. b
8. c
9. c
10. b
11. c
12. c
13. c
14. c
15. b
16. c
17. b
18. b
19. b
20. c
21. b
22. b
23. c
24. b
25. c

Penutup

Selamat! Anda telah menyelesaikan panduan dasar mempelajari Arduino ini. Anda kini memiliki pemahaman tentang apa itu Arduino, komponen elektronik dasar, cara memprogramnya menggunakan bahasa C/C++, mengontrol output seperti LED, membaca input dari tombol, berinteraksi dengan sensor umum, dan menampilkan informasi ke LCD.

Perjalanan Anda dengan Arduino baru saja dimulai. Dunia elektronika dan mikrokontroler sangat luas dan penuh dengan kemungkinan menarik. Jangan ragu untuk terus bereksperimen, mencoba proyek-proyek baru, dan mencari sumber belajar tambahan dari komunitas Arduino yang luas. Semakin banyak Anda berlatih, semakin mahir Anda dalam mewujudkan ide-ide kreatif Anda.

Teruslah belajar dan berkarya!

Referensi

Arduino. (2025, January 1). *Arduino Official Store*. Retrieved from Arduino Official Store: https://store.arduino.cc/

Autodesk. (2025, January 1). *Instrucable*. Retrieved from Instrucable: https://www.instructables.com/Arduino-How-to-Control-Servo-Motor-With-Potentiome/

ThinkerCad. (2025, January 1). *ThinkerCad.com*. (AutoDesk) Retrieved January 1, 2025, from https://www.tinkercad.com/

Wokwi. (2019, January 1). *wokwi.com*. Retrieved from wokwi.com: wokwi.com

Tentang Penulis

Eko Travada Suprapto Putro, S.T., M.T. adalah seorang praktisi dan akademisi di bidang teknik komputer dan mikrokontroler. Ia menyelesaikan pendidikan sarjana Teknik Elektro di Universitas Kristen Maranatha pada tahun 1998 dan meraih gelar Magister Teknik dari Institut Teknologi Bandung pada tahun 2004 dengan konsentrasi Teknik Komputer.

Memiliki keahlian dalam pemrograman C, Python, pengembangan sistem berbasis Arduino dan Atmel, serta pengelolaan basis data MySQL. Pengalamannya yang luas mencakup berbagai proyek sistem informasi militer dan sistem deteksi plagiarisme. Di antaranya sebagai konsultan ahli untuk Kodam II/Sriwijaya, Kodam IV/Diponegoro, dan Seskoad.

Saat ini, beliau aktif menulis, mengajar, dan mengembangkan panduan teknis berbasis mikrokontroler untuk pemula hingga tingkat lanjutan.